Displaying results of processing power monitoring and analysis of parallel processing system

Publication number: DE19710252
Publication date: 1998-02-26
Inventor: ASAI NOBORU (JP); MATSUMOTO TOHRU (JP); WATANABE KAZUO (JP)
Applicant: FUJITSU LTD (JP)
Classification:
- international: ***G06F9/06; G06F11/32; G06F11/34; G06F15/16; G06F15/177;*** **G06F9/06; G06F11/32; G06F11/34; G06F15/16;** (IPC1-7): G06F11/34
- European: G06F11/32P; G06F11/34B
Application number: DE19971010252 19970313
Priority number(s): JP19960222012 19960823

Also published as:
US5903730 (A1)
JP10063550 (A)

Abstract of **DE19710252**

The processing system 11 has a number of parallel processors 12. Instruction routine execution data is collected 13 and used to generate a processing profile in the form of a series of reports. This allows the various routines to be classified. The execution time data is summed 14 and a statistical evaluation made. A graphical representation is made that shows maximum, minimum, average and standard variation of the execution times of the routines.

(19) BUNDESREPUBLIK DEUTSCHLAND

DEUTSCHES PATENTAMT

(12) **Offenlegungsschrift**

(10) **DE 197 10 252 A 1**

(51) Int. Cl.[6]: G 06 F 11/34

(21) Aktenzeichen: 197 10 252.2
(22) Anmeldetag: 13. 3. 97
(43) Offenlegungstag: 26. 2. 98

DE 197 10 252 A 1

(30) Unionspriorität:

P 8-222012 23.08.96 JP

(71) Anmelder:

Fujitsu Ltd., Kawasaki, Kanagawa, JP

(74) Vertreter:

W. Seeger und Kollegen, 81369 München

(72) Erfinder:

Asai, Noboru, Shibuoka, Shizuoka, JP; Matsumoto, Tohru, Shibuoka, Shizuoka, JP; Watanabe, Kazuo, Numazu, Shizuoka, JP

Prüfungsantrag gem. § 44 PatG ist gestellt

(54) Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -Analyse in einem Parallelrechnersystem

(57) Es wird ein Verfahren der sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse für ein Parallelrechnersystem beschrieben, bei dem eine Vielzahl von Prozessoren ein parallel verarbeitendes Programm ausführen, welches aus einer Vielzahl von Routinen zusammengesetzt ist. Zuerst werden Informationen hinsichtlich der Ausführungszeit von jeder Routine in einer Realzeitweise gesammelt, während das Programm gleichlaufend durch die Vielzahl von Prozessoren ausgeführt wird. Zweitens werden Maximal-, Mittel- und Minimalwerte der Ausführungszeit von jeder Routine berechnet, basierend auf den Informationen, die für die Vielzahl der Prozessoren gesammelt wurden. Drittens werden die gesammelten Informationen in Form eines Ausführungsprofils zusammengefaßt oder summiert und werden in graphischer Form dargestellt, und zwar unter Verwendung von Balkengraphiken, Kreisgraphiken oder Radarkarten. Für jede Prozedur oder Programmschleife klassifiziert das vorliegende Verfahren die Prozentsätze der Anwender-Netto-Programmausführung, Kommunikation, Synchronisation und anderer parallel laufender Zusatzaufgaben, und zeigt auch deren Abweichungen an. Dieses Merkmal unterstützt den Anwender darin, das aktuelle Programmverhalten zu verstehen und das parallel verarbeitende Programm abzustimmen.

DE 197 10 252 A 1

Beschreibung

HINTERGRUND DER ERFINDUNG

1. Gebiet der Erfindung

Die vorliegende Erfindung betrifft ein Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse in einem Parallelrechnersystem, welches eine Vielzahl von Prozessoren verwendet, und betrifft spezieller ein Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse in einer bedienungskomfortableren und umfassenden Weise. Die vorliegende Erfindung betrifft auch ein computer-lesbares Medium, welches mit einem Computerprogramm codiert ist, bei welchem das zuvor genannte Verfahren realisiert wird.

2. Beschreibung des verwandten Standes der Technik

Parallele Verarbeitungstechniken spielen eine entscheidende Rolle bei der Befriedigung von heutzutage wachsenden Anforderungen nach Hochleistungsrechnersystemen. Ein Parallelrechnersystem führt eine Anwendung aus, indem es eine Vielzahl von Prozessoren verwendet, die parallel laufen, wodurch ein hoher Verarbeitungsdurchgang erzielt wird. Die Entwicklung von Software für ein Parallelrechnersystem bringt jedoch im allgemeinen einige Schwierigkeiten bei der Programmaustestung, Bewertung und Leistungsabstimmung mit sich, und zwar aufgrund deren komplexer Natur.

Um das Problem zu lösen, werden Leistungsanalysierer als ein günstiges Werkzeug eingeführt, welches die Software-Programmierer darin unterstützt, das Verhalten eines Parallelprogramms beim Testen zu überwachen und dessen Leistung zu messen. Der Analysierer verarbeitet die gesammelten Daten und präsentiert das Ergebnis dem Anwender in Form eines aufsummierten Ausführungsprofils.

Ein herkömmlicher Leistungsanalysierer erzeugt ein Ausführungsprofil von einzelnen Prozessoren, welches in solcher Weise aufsummiert ist, daß die gesamte Verarbeitungszeit in einige Kategorien klassifiziert ist, enthaltend Netto-Betriebszeit und Leerlaufzeit usw. Das Ausführungsprofil jedes Prozessors wird getrennt in Form einer Balkengraphik oder ähnlichem dargestellt.

Dieses herkömmliche Verfahren der Darstellung von Ausführungsprofilen ist jedoch für Software-Entwicklungsingenieure in keiner Weise vorteilhaft, wenn das Parallelverarbeitungssystem hunderte von Prozessoren enthält. Die an einem Analysierer-Bildschirm dargestellten Informationen sind zu umfangreich, um das Programmverhalten richtig verstehen zu können, da zu viele Prozessoren parallel laufen. In einem solchen Fall kann der herkömmliche Analysierer die Schwierigkeit dadurch etwas mildern, indem eine gewisse statistische Verarbeitung hinsichtlich der gewonnen Rohdaten angewandt wird. Dies erlaubt es, das Gesamtprogrammverhalten oder die mittlere Leistung der Prozessoren den Ingenieuren in einer verständlicheren Art darzubieten.

Jedoch sind die Informationen, die durch diesen herkömmlichen Leistungsanalysierer vorgesehen werden, für die Software-Entwicklungsingenieure nicht ausreichend, um die Systemqualität bzw. Leistung vollständig verstehen zu können und die detaillierte Beziehung zwischen den Prozessen verstehen zu können, die parallel laufen.

ZUSAMMENFASSUNG DER ERFINDUNG

Unter Einbeziehung des oben gesagten, besteht eine erste Aufgabe der vorliegenden Erfindung darin, ein verbessertes Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse in einem Parallelrechnersystem zu schaffen, welches Prozentsätze der Anwender-Netto-Programmausführung, Kommunikation, Synchronisation und anderer parallel ablaufender Organisationsaufgaben zeigt als auch deren Abweichungen anzeigt.

Es ist ein weiteres Ziel der vorliegenden Erfindung, ein verbessertes Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse in einem Parallelrechnersystem zu schaffen, welches dem Anwender die Möglichkeit bietet, Software-Routinen in einem parallel verarbeitenden Programm zu bewerten, und zwar über eine problemlose und vorteilhafte graphische Wiedergabe des Ausführungsprofils.

Um die genannte Aufgabe zu lösen, schafft die vorliegende Erfindung ein Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse in einem Parallelrechnersystem, bei dem eine Vielzahl von Prozessoren ein paralleles Verarbeitungsprogramm ausführen, welches aus einer Vielzahl von Routinen zusammengesetzt ist. Das Verfahren umfaßt die folgenden Schritte:

(a) Sammeln von Informationen hinsichtlich der Ausführungszeit von jeder Routine, die gleichlaufend durch eine Vielzahl von Prozessoren ausgeführt wird, unter Berücksichtigung der Klassifizierung der Routinen;
(b) Ableiten eines Maximalwertes, eines Mittelwertes, eines Minimalwertes und einer Standardabweichung der Ausführungszeit von jeder Routine basierend auf den Informationen, die für die Vielzahl der Prozessoren gesammelt wurden; und
(c) graphisches Darstellen der Ausführungsprofil-Information, welche den Maximalwert, den Mittelwert und den Minimalwert und die Standardabweichung der Ausführungszeit jeder Routine enthält.

Um speziell das weitere Ziel, welches oben erläutert wurde, zu erreichen, wird ein Verfahren zur sichtbaren Darstellung von Ergebnissen der Leistungsüberwachung und -analyse für ein Parallelrechnersystem geschaffen.

Gemäß dem Verfahren werden Ausführungsprofil-Informationen in Form von Balkengraphiken, Kreisgraphiken oder Radarkarten dargestellt.

Weitere Ziele neben den obigen, Merkmale und Vorteile der vorliegenden Erfindung ergeben sich aus der folgenden Beschreibung in Verbindung mit den beigefügten Zeichnungen, die eine bevorzugte Ausführungsform der vorliegenden Erfindung als Beispiel veranschaulichen.

KURZBESCHREIBUNG DER ZEICHNUNGEN

Fig. 1 ist eine Konzeptdarstellung eines Leistungsanalysierersystems, welches ein Sichtdarstellungsverfahren gemäß der vorliegenden Erfindung verwendet;

Fig. 2 ist ein Flußdiagramm, welches einen Prozeß der Modifizierung und Kompilierung eines Programms zeigt, welches für ein Parallelverarbeitungssystem geschrieben wurde;

Fig. 3 ist eine Tabelle, in der Laufzeit-Subroutinen aufsummiert sind, die für eine Leistungsüberwachung und -analyse vorgesehen werden;

Fig. 4 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #1 beschreibt;

Fig. 5 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #2 beschreibt;

Fig. 6 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #3 beschreibt;

Fig. 7 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #4 beschreibt;

Fig. 8 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #5 beschreibt;

Fig. 9 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #6 beschreibt;

Fig. 10 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #7 beschreibt;

Fig. 11 ist ein Flußdiagramm, welches die Funktionen der Laufzeit-Subroutine #8 beschreibt;

Fig. 12 ist eine Tabelle, in der die Informationen aufsummiert sind, die durch das Leistungsanalysierersystem der vorliegenden Erfindung zu sammeln sind;

Fig. 13 ist ein Diagramm, welches ein erstes Beispiel einer Ausführungsprofilanzeige gemäß der vorliegenden Erfindung zeigt, bei der jede Prozedur oder Schleife skaliert ist in bezug auf die Ausführungszeit eines gesamten Programms;

Fig. 14 ist ein Diagramm, welches ein zweites Beispiel der Ausführungsprofildarstellung zeigt, bei der alle Prozeduren und Schleifen in gleicher Weise skaliert sind, ungeachtet deren absoluter Ausführungszeiten;

Fig. 15 ist ein Diagramm, welches ein drittes Beispiel einer Ausführungsprofildarstellung zeigt, in der das Ausführungsprofil von jeder Prozedur oder Schleife in einer Kreisgraphik wiedergegeben ist;

Fig. 16 ist ein Diagramm, welches ein viertes Beispiel der Ausführungsprofildarstellung zeigt, in der das Ausführungsprofil in einer Radarkarte wiedergegeben ist;

Fig. 17 ist ein Diagramm, welches ein fünftes Beispiel einer Ausführungsprofildarstellung gemäß der vorliegenden Erfindung zeigt; und

Fig. 18 ist ein Diagramm, welches ein sechstes Beispiel der Ausführungsprofildarstellung gemäß der vorliegenden Erfindung zeigt.

BESCHREIBUNG DER BEVORZUGTEN AUSFÜHRUNGSFORM

Eine Ausführungsform der vorliegenden Erfindung soll nun im folgenden unter Hinweis auf die beigefügten Zeichnungen beschrieben werden.

Fig. 1 ist eine Konzeptdarstellung eines Leistungsanalysierersystems, in welchem ein Sichtbardarstellverfahren gemäß der vorliegenden Erfindung angewandt wird. Ein Parallelrechnersystem 11 ist durch eine Vielzahl von Prozessoren 12a—12n aufgebaut, die als Berechnungselemente dienen und die zusammen laufen, um umfangreiche Berechnungsaufgaben zu erzielen, die in einem Parallelverarbeitungsprogramm codiert sind. Solche Prozessoren 12a—12n umfassen einzeln Datensammeleinrichtungen 13a—13n und eine Datensummiereinrichtung 14a—14n zum Korrigieren und Aufsummieren von jeweils Ausführungszeitdaten.

Die Datensammeleinrichtungen 13a—13n sammeln in einer Real zeitweise die Ausführungszeitdaten während der Programmausführung in den jeweiligen Prozessoren 12a—12n, in denen sie auftreten. Die Ausführungszeitdaten, die durch die Datensammeleinrichtungen 13a—13n gesammelt wurden, werden als Quellendaten für die Ausführungsprofile verwendet. Die Ausführungsprofile bestehen aus einem Satz von Reports, welche die Anwender mit Informationen über die Prozessoren 12a—12n, daß diese tatsächlich arbeiten, versehen, wodurch diese das Programm von verschiedenen Gesichtspunkten aus bewerten können.

Die Ausführungsprofile weisen auf einige bestimmte Programmroutinen hin, welche das Parallelverarbeitungsprogramm bilden, wie beispielsweise Prozeduren, Schleifen, eine parallele Verarbeitungsbibliothek, die durch das Leistungsanalysierersystem überwacht und analysiert werden. In diesem Sinn werden diese Routinen als Subjekte der Profilbildung oder Profilsubjekte bezeichnet.

Die Datensummiereinrichtungen 14a—14n summieren die Ausführungszeitdaten, die durch die Datensammeleinrichtungen 13a—13n gesammelt wurden. Spezifisch gesagt, berechnen sie Maximal-, Mittel-, Minimalwerte und eine Standardabweichung der gesammelten Ausführungszeiten für jede Routine oder jedes Subjekt für die Profilbildung. Um diese Aufgabe zu erreichen, ist es erforderlich, alle Ausführungszeitdaten zu kompilieren, die durch die Datensammeleinrichtungen 13a—13n an einer einzelnen Stelle gesammelt wurden. Daher befaßt sich eine der Datensummiereinrichtungen, wie beispielsweise 14a, mit den Summierungsaufgaben und andere 14b—14n senden einfach ihre Daten zu dieser Datensummier-Subroutine 14a unter Verwendung von Inter-Prozessor-Kommunikationspfaden (nicht gezeigt) in dem Parallelrechnersystem 11.

In Wirklichkeit sind die oben beschriebenen Datensammeleinrichtungen 13a—13n und die Datensummiereinrichtungen 14a—14n in dem Parallelrechnersystem 11 in Form eines Satzes von Subroutinen implementiert, die

als ein Teil der Laufzeitbibliothek für die Parallelverarbeitung vorgesehen sind. Bei der vorliegenden Ausführungsform stehen acht Laufzeit-Subroutinen für diesen Zweck zur Verfügung, wie in Fig. 3 aufgelistet ist. Die detaillierten Funktionen der Datensammeleinrichtungen 13a—13n und der Datensummiereinrichtungen 14a—14n sollen getrennt unter Hinweis auf die Fig. 4 bis 11 beschrieben werden.

Um erneut auf Fig. 1 einzugehen, ist das Parallelrechnersystem 11 an zwei Datendateien gekoppelt: eine Quellencodedatei 15 und eine Profilinformationsdatei 16. Die Quellencodedatei 15 enthält den originalen Quellencode eines Programms, welches durch die Prozessoren 12a—12n ausgeführt werden soll. Die Profilinformationsdatei 16 speichert Ausführungszeitdaten und deren Summendaten, wie sie von den Prozessoren 12a—12n benötigt werden.

Ein Profilanzeigeprozessor 17 analysiert die Inhalte der Profilinformationsdatei 16 und stellt das Ergebnis dieser Analyse in einem Ausführungsprofilfenster 22 auf einem Bildschirm einer Anzeigeeinheit 21 dar. Hier macht der Profilanzeigeprozessor 17 das Ergebnis der Analyse sichtbar, und zwar unter Verwendung mehrerer graphischer Wiedergabetechniken, wie dies an späterer Stelle in Verbindung mit den Fig. 13 bis 18 beschrieben werden soll. Eine Eingabevorrichtung 18 wird dazu verwendet, um einige Befehle oder Daten in den Profilanzeigeprozessor 17 einzugeben.

Bei Verwendung dieser Eingabevorrichtung 18 kann der Anwender dieses Leistungsanalysierersystems einen spezifischen Punkt (z. B. eine Prozedur oder Schleife) bezeichnen, der in dem Ausführungsprofilfenster 22 dargestellt wird. Ein Quellencode-Positionsanalysierer 19 sucht nach dem Quellencode, um die Codeposition (oder Zeilenzahl) entsprechend dem Punkt herauszufinden, der durch den Anwender bezeichnet worden ist. Gemäß der Codeposition-Information, die durch den Quellencode-Positionsanalysierer 19 geliefert wird, extrahiert einen Quellencode-Browser 20 den relevanten Teil des Quellencodes 15 und zeigt den extrahierten Quellencode in einem Quellencodefenster 23 auf dem Bildschrift der Anzeigeeinheit 21 an.

Bevor die Programmausführung bei dem oben beschriebenen System gestartet wird, ist es erforderlich, den originalen Quellencode in ein ausführbares Programm zu übersetzen oder zu kompilieren. Bei der vorliegenden Erfindung sollte der Quellencode vor der Kompilierung modifiziert werden, und zwar für Überwachungs- und Meßzwecke. Spezifischer gesagt, schiebt das Kompiliersystem Subroutine-Ruf-Zustände in den Quellencode ein, um einige Laufzeit-Subroutinen für die Datensammlung und Summierung an geeigneten Punkten in dem Programm aufzurufen.

Der Quellencode eines parallel verarbeitenden Programms besitzt eine hierarchische Struktur. Das Programm oder die Anwender-Anwendung besteht aus einer Vielzahl von Prozeduren. Eine Prozedur ruft einige andere Prozeduren auf; mit anderen Worten können Prozeduren verschachtelt werden. Als Tatsache ist der Hauptkörper des Programms eine Prozedur, die implizit als "MAIN" bezeichnet wird, die mehrere Abkömmlingsprozeduren enthält. Die Schleife besteht aus einer Sequenz von Befehlen als Teil einer Prozedur, die iterativ ausgeführt werden.

Um die paralle Programmausführung zu unterstützen, sieht das System eine Bibliothek von Routinen vor, um eine Prozeßkommunikation, Synchronisation und andere Aufgaben auszuführen, die bei der parallelen Verarbeitung inhärent vorkommen. Diese werden als parallel verarbeitende Bibliotheksroutinen bezeichnet, welche durch die Prozeduren nach Bedarf aufgerufen werden. Da sie die Systemleistung als parallel laufende Zusätze beeinflussen, sind die parallel verarbeitenden Bibliotheksroutinen ein wichtiges Merkmal bei der Ausführung der Profilbildung.

Fig. 2 zeigt ein Flußdiagramm, welches einen Prozeß der Modifizierung und Kompilierung eines Programms für das Parallelrechnersystem 11 zeigt. Dieser Prozeß enthält die folgenden sieben Schritte.

[S1] Es wird der Quellencode eines parallel verarbeitenden Programms, welches in jedem Prozessor in dem Parallelrechnersystem 11 auszuführen ist, in das Kompilierungssystem eingegeben.

[S2] Es wird eine Feststellung zum Aufrufen einer Laufzeit-Subroutine #1 in den Anfangsteil des Hauptprogrammkörpers eingefügt. Die Laufzeit-Subroutine #1 zeichnet die Zeit auf, wenn das Programm gestartet wird.

[S3] Es werden Feststellungen zum Aufrufen der Laufzeit-Subroutinen #2 und #3 jeweils in den Anfangsteil und in den Endteil jeder Prozedur eingefügt, die in dem Quellencode enthalten ist.

[S4] Es werden Feststellungen zum Aufrufen der Laufzeit-Subroutinen #4 und #5 jeweils in den Anfangsteil und in den Endteil jeder Schleife eingefügt, die in den Prozeduren enthalten ist.

[S5] Es werden Feststellungen zum Aufrufen der Laufzeit-Subroutinen #6 und #7 jeweils in den Anfangsteil und in den Endteil jeder parallel verarbeitenden Bibliotheksroutine eingefügt.

[S6] Es wird eine Feststellung zum Aufrufen der Laufzeit-Subroutine #8 in den Endteil des Hauptprogrammkörpers eingefügt.

[S7] Das Kompilierungssystem übersetzt (oder kompiliert) die Quellencodedatei 15, die nun die bei den Schritten S2 bis S6 eingefügten Subroutine-Aufruf-Feststellungen enthält, in den Objektcode, der durch die Prozessoren 12a—12n ausführbar ist.

Durch Ausführen der oben erläuterten Schritte S1 bis S7 fügt das Kompilierungssystem einige geeignete Feststellungen in die Quellencodedatei 15 ein, um die Laufzeit-Subroutinen für die Datensammlung und Aufsummierung aufzurufen und es kompiliert die Quellencodedatei 15, um den ausführbaren Objektcode zu generieren. Als ein Ergebnis enthält der generierte Objektcode Subroutine-Aufruf-Befehle sowohl an den Start- als auch Endpunkten des Hauptprogrammkörpers, der Prozeduren, Schleifen und der parallel verarbeitenden Bibliotheksroutinen.

Fig. 3 faßt die Laufzeit-Subroutinen, die zum Zweck der Leistungsüberwachung und -analyse eingeführt wurden, zusammen. Die Tabelle 30 beschreibt kurz die Funktionen der acht Laufzeit-Subroutinen in deren jeweiligen Gelegenheiten, bei denen sie aufgerufen werden.

Die in dieser Tabelle aufgelisteten ersten sieben Subroutinen (#1 bis #7) sind Subroutinen, die zum Sammeln von Zeitwertzeichendaten und zum Akkumulieren der Ausführungszeit von jedem Abschnitt des Parallelpro-

gramms ausgelegt sind. Das heißt, solche Subroutinen #1—#7 entsprechend der Datensammeleinrichtung 13a—13n in Fig. 1. Die achte Subroutine #8 besteht aus einer Subroutine, die zum Summieren der Daten ausgelegt ist, die durch die Subroutinen #1 bis #7 gesammelt und akkumuliert wurden. Die Subroutine #8 entspricht somit der Datensummiereinrichtung 14a—14n.

Um auf die Fig. 4 bis 11 einzugehen, so das folgende hinsichtlich der detaillierten Funktionen der Laufzeit-Subroutinen #1 — #8 erläutert, die in den Quellencode eingefügt sind.

Fig. 4 zeigt ein Flußdiagramm, welches die Laufzeit-Subroutine #1 zeigt, die dann aufgerufen wird, wenn das Programm startet zu laufen und die zwei folgenden Aufgaben vorsieht.

[S11a] Initialisieren eines örtlichen Datenbereiches, der zum Aufzeichnen und zum Analysieren des Prozessor-Verhaltens verwendet wird.

[S11b] Aufzeichnen des Programmstartzeitpunktes.

Die Fig. 5 und 6 sind Flußdiagramme, welche die Laufzeit-Subroutinen #2 bzw. #3 zeigen. Die Laufzeit-Subroutine #2 wird zu Beginn von jeder Prozedur aufgerufen und wie bei dem Schritt S12a von Fig. 5 gezeigt ist, zeichnet sie den Ausführungsstartzeitpunkt dieser Prozedur auf. Als ein Gegenstück zu dieser Subroutine #2 wird die Laufzeit-Subroutine #3 aufgerufen, wenn die Prozedur beendet wird und sie führt die folgenden drei Aufgaben aus:

[S13a] Erhalten der Ausführungs-Ende-Zeit der Prozedur.

[S13b] Akkumulieren der Ausführungszeit der Prozedur (das heißt Akkumulieren der Differenz zwischen der Ausführungsstartzeit und der Ausführungs-Ende-Zeit).

[S13c] Akkumulieren der Ausführungszeit von jeder parallel verarbeitenden Bibliotheksroutine, die durch die Prozedur aufgerufen wurde (das heißt Akkumulieren der Differenz zwischen der Ausführungsstartzeit und der Ausführung-Ende-Zeit, die durch die Laufzeit-Subroutinen #6 und #7 für jede parallel verarbeitende Bibliotheksroutine aufgezeichnet wurde).

Fig. 7 zeigt ein Flußdiagramm der Laufzeit-Subroutine #4, welche aufgerufen wird, wenn der Prozessor in eine Schleife eintritt. Die Schleife umfaßt eine Folge von Befehlen, die iterativ ausgeführt werden, und zwar unter der Steuerung einer Schleifen-Zähler-Variablen. Jedesmal, wenn die Schleife ausgeführt wird, wird die Schleifen-Zähler-Variable erhöht (oder vermindert), und zwar um ein vorbestimmtes Inkrement und es wird die Schleife wiederholt, bis die Schleifen-Zähler-Variable einen vorgeschriebenen Grenzwert erreicht oder andere Beendigungsbedingungen erfüllt sind. Um statistische Ausführungsdaten für die Schleifen zu sammeln, führt die Laufzeit-Subroutine #4 die folgenden zwei Aufgaben durch:

[S14a] Aufzeichnen der Ausführungsstartzeit der Schleife.

[S14b] Erhalten des Anfangswertes der Schleifen-Zähler-Variablen und des Inkrements (oder Schleifenzähler-Schrittgröße) für jede Iteration, und Aufbewahren derselben für zukünftige Verwendung.

Fig. 8 zeigt ein Flußdiagramm der Laufzeit-Subroutine #5, die ausgeführt wird, wenn eine Schleife beendet wird. Diese füllt die folgenden vier Aufgaben für jede Schleife.

[S15a] Erhalten der Ausführung-Ende-Zeit der Schleife.

[S15b] Akkumulieren der Ausführungszeit der Schleife (das heißt Akkumulieren der Differenz zwischen der Ausführungsstartzeit und der Ausführung-Ende-Zeit für die Schleife).

[S15c] Erhalten des endgültigen Wertes der Schleifensteuer-Variablen und Berechnen der Zahl der Iterationen basierend auf den Anfangs- und Endwerten der Schleifen-Zähler-Variablen und der Schrittgröße.

[S15d] Akkumulieren der Zahl der Iterationen.

[S15e] Akkumulieren der Ausführungszeit von jeder parallel verarbeitenden Bibliotheksroutine, die durch die Schleife aufgerufen wird. Hier wird die Ausführungszeit dadurch erhalten, indem die Ausführungsstartzeit bzw. Ausführungsstartzeitpunkt von dem Ausführung-Ende-Zeitpunkt subtrahiert wird, die durch die Laufzeit-Subroutinen #6 und #7 für jede parallel verarbeitende Bibliotheksroutine aufgezeichnet wurden, die von der Schleife aufgerufen wurden.

Die Fig. 9 und 10 zeigen die Laufzeit-Subroutinen #6 und #7, die aufgerufen werden, wenn eine parallel verarbeitende Bibliotheksroutine jeweils startet und endet. Das heißt, die Laufzeit-Subroutine #6 führt den folgenden Schritt aus:

[S16a] Aufzeichnen des Ausführungsstartzeitpunktes der parallel verarbeitenden Bibliotheksroutine.

Als Gegenstück zu dieser Laufzeit-Subroutine #6 erzeugt die Laufzeit-Subroutine #7 folgendes:

[S17a] Erhalten des Ausführungs-Ende-Zeitpunktes der parallel verarbeitenden Bibliotheksroutine.

[S17b] Akkumulieren der Ausführungszeit der parallel verarbeitenden Bibliotheksroutine, wobei die Ausführungszeit die Differenz zwischen dem Ausführungsstartzeitpunkt, der bei dem Schritt S16a aufgezeichnet wurde, und dem Ausführung-Ende-Zeitpunkt, der bei dem Schritt S17a erhalten wurde, ist.

Fig. 11 zeigt ein Flußdiagramm, welches die Laufzeit-Subroutine #8 wiedergibt, die am Ende des gesamten Programms ausgeführt wird. Die Laufzeit-Subroutine #8 summiert die Ausführungszeitdaten auf, die durch die anderen Laufzeit-Subroutinen #1 bis #7 gesammelt wurden, und zwar gemäß den folgenden Schritten:

[S18a] Berechnen der mittleren angesammelt Zeit von jeder parallel verarbeitenden Bibliotheksroutine, die durch jede Prozedur aufgerufen wurde, gemäß den folgenden Gleichungen:

$$Tcp(avg) = \frac{1}{n}\sum_{m=1}^{n} Tcp(m) \qquad \ldots(1)$$

$$Twp(avg) = \frac{1}{n}\sum_{m=1}^{n} Twp(m) \qquad \ldots(2)$$

$$Top(avg) = \frac{1}{n}\sum_{m=1}^{n} Top(m) \quad \ldots(3)$$

worin n eine ganze Zahl ist, welche die Anzahl der Prozessoren wiedergibt, die in dem Parallelrechnersystem 11 involviert sind und m eine ganze Zahl ist, die von 1 bis n reicht und dazu verwendet wird, um einen spezifischen Prozessor anzuzeigen. Tcp(m) ist die akkumulierte Ausführungszeit, welche der m'te Prozessor für die Kommunikationsaufgaben verbraucht hat. Tcp(avg) bedeutet eine mittlere angesammelte Ausführungszeit, die für Kommunikationsaufgaben verbraucht wurde. Twp(m) ist die akkumulierte Ausführungszeit, die der m'te Prozessor für die Synchronisation verbraucht hat. Twp(avg) bedeutet eine mittlere akkumulierte Ausführungszeit, die für Synchronisationsaufgaben verbraucht wurde. Top(m) ist die akkumulierte Ausführungszeit, die der m'te Prozessor für Aufgaben verbraucht hat, die anders sind als die Kommunikations- und Synchronisationsaufgaben. Top(avg) bedeutet eine mittlere akkumulierte Ausführungszeit, die für andere Aufgaben verbraucht wurde.

[S18b] Berechnen des Maximalwertes, Minimalwertes und der Standardabweichung σ für jede der akkumulierten Ausführungszeiten Tcp(m), Twp(m) und Top(m).

[S18c] Berechnen der mittleren akkumulierten Ausführungszeit des Anwenderprogramms gemäß der folgenden Gleichung:

$$Tup(avg) = \frac{1}{n}\sum_{m=1}^{n} Tup(m)$$

$$= \frac{1}{n}\sum_{m=1}^{n} (Tpp(m) - Tcp(m) - Twp(m) - Top(m)) \quad \ldots(4)$$

worin Tpp(m) die Ausführungszeit bedeutet, die durch den m'ten Prozessor für die in Betracht stehende Prozedur verbraucht wurde und wobei Tup(m) die Netto-Ausführungszeit des Anwenderprogramms bedeutet.

$$Tup(m) - Tpp(m) - Tcp(m) - Twp(m) - Top(m)) \quad (5)$$

[S18d] Berechnen des Maximalwertes, des Minimalwertes und der Standardabweichung σ für die akkumulierte Ausführungszeit des Anwenderprogramms Tup(m).

[S18e] Berechnen der mittleren angesammelten Zeit von jeder parallel verarbeitenden Bibliotheksroutine, die in jeder Schleife aufgerufen wurde, gemäß den folgenden Gleichungen:

$$Tcl(avg) = \frac{1}{n}\sum_{m=1}^{n} Tcl(m) \quad \ldots(6)$$

$$Twl(avg) = \frac{1}{n}\sum_{m=1}^{n} Twl(m) \quad \ldots(7)$$

$$Tol(avg) = \frac{1}{n}\sum_{m=1}^{n} Tol(m) \quad \ldots(8)$$

Tcl(m) ist die akkumulierte Ausführungszeit, die der m'te Prozessor für Kommunikationsaufgaben benötigt hat. Tcl(avg) bedeutet die mittlere angesammelte Ausführungszeit, die für Kommunikationsaufgaben verbraucht wurde. Twl(m) ist die akkumulierte Ausführungszeit, die der m'te Prozessor für die Synchronisation benötigt hat. Twl(avg) bedeutet die mittlere akkumulierte Ausführungszeit, die für Synchronisationsaufgaben verbraucht wurde. Tol(m) ist die akkumulierte Ausführungszeit, die der m'te Prozessor für Aufgaben verbraucht hat, anders als die Kommunikations- und Synchronisationsaufgaben. Tol(avg) bedeutet die mittlere akkumulierte Ausführungszeit, die für andere Aufgaben verbraucht wurde.

[S18f] Berechnen des Maximalwertes, des Minimalwertes und der Standardabweichung σ für jede der akkumulierten Ausführungszeiten Tcl(m), Twl(m) und Tol(m).

[S18g] Berechnen der mittleren angesammelten Ausführungszeit des Anwenderprogramms gemäß der folgenden Gleichung:

$$Tul(ave) = \frac{1}{n}\sum_{m=1}^{n} Tul(m)$$

$$= \frac{1}{n}\sum_{m=1}^{n} (Tpl(m) - Tcl(m) - Twl(m) - Tol(m)) \quad \ldots(9)$$

worin Tpl(m) die Ausführungszeit ist, die von dem m'ten Prozessor für die in Betracht stehende Schleife verbraucht wurde und Tul(m) die akkumulierte Netto-Ausführungszeit des Anwenderprogramms ist.

$$Tul(m)-Tpl(m)-Tcl(m)-Twl(m)-Tol(m)) \quad (5)$$

[S18h] Berechnen des Maximalwertes, des Minimalwertes und der Standardabweichung σ für die akkumulierte Ausführungszeit des Anwenderprogramms Tul(m).

[S18i] Ausgeben der Ergebnisse der Schritte S18a bis S18h als eine Profil-Informationsdatei **16**.

Fig. 12 zeigt eine Tabelle, die die Informationen zusammenfaßt, welche durch das Leistungsanalysierersystem der vorliegenden Erfindung gesammelt wurden. In dieser Tabelle **40** sind die gesammelten Informationen in Einklang mit dem Subjekt der Profilierungsoperationen klassifiziert, welche Prozeduren, Schleifen und parallel verarbeitende Bibliotheksroutinen enthalten, um die Kommunkation, Synchronisation und andere Aufgaben zu unterstützen. Obwohl in **Fig. 12** nicht gezeigt, könnten bedingte Programmverzweigung auch in dem Profilsubjekt enthalten sein.

Im allgemeinen werden Informationen, welche das Programmverhalten in jedem Prozessor betreffen, an drei verschiedenen Stufen der Ausführung gesammelt: einer vorbereitenden Verarbeitungsstufe, einer Anlauf-Verarbeitungsstufe und einer Datenverarbeitungsstufe (terminal process stage).

Die vorbereitende Verarbeitungsstufe besteht aus einer Stufe, bevor eine bezeichnete Routine als Profilsubjekt aufgerufen wird. Die drei Arten der Profil-Subjekte, die in der Tabelle **40** aufgelistet sind, erfordern es jedoch nicht, daß Informationen an der vorbereitenden Verarbeitungsstufe gesammelt werden. Wie oben dargelegt wurde, können auch bedingte Programmverzweigungen (in **Fig. 7** nicht gezeigt) ein Ziel der Ausführungsprofilierung sein. Wenn dies der Fall ist, werden zusätzliche Befehle in den Quellencode eingeführt, um eine andere Laufzeit-Subroutine aufzurufen, um Informationen an aktuellen Verzweigungsbedingungen zu sammeln. Speziell enthalten solche Verzweigungsbedingungen "Wahr- oder Falsch-"Bedingungen als Ergebnis von logischen Operationen oder "negative, Null- oder positive" Bedingungen als ein Ergebnis von arithmetischen Operationen, die vor den Verzweigungsbefehlen ausgeführt wurden. Die Laufzeit-Subroutine zählt das Auftreten von jeder solcher Bedingungen, bevor eine Verzweigung durchgeführt wird, nämlich bei der vorbereitenden Verarbeitungsstufe.

Die Anlauf-Verarbeitungsstufe besteht aus einer Stufe, bei der eine Routine, die bei einem Profil-Subjekt beteiligt ist, zu laufen beginnt. Spezieller gesagt, wird die Ausführungsstartzeit an der Anlauf-Verarbeitungsstufe von jeder Prozedur (siehe S12a in **Fig. 5**), Schleife (siehe S14a in **Fig. 7**) oder parallel verarbeitenden Bibliotheksroutine (siehe S16a in **Fig. 9**) aufgezeichnet. Im Falle der Schleifen wird der Anfangswert einer Schleifenzähler-Variablen und einer Schleifenzähler-Schrittgröße zusätzlich zu der Ausführungsstartzeit (siehe S14b in **Fig. 7**) aufgezeichnet.

Die Datenverarbeitungsstufe oder Ende-Verarbeitungsstufe besteht aus einer Stufe, bei der eine Routine dabei ist zu enden. Bei der Ende-Verarbeitungsstufe jeder Prozedur werden die Ausführungszeit der Prozedur und die Ausführungszeit von jeder parallel verarbeitenden Bibliotheksroutine, die innerhalb der Prozedur aufgerufen wurde, akkumuliert (siehe S13b und S13c in **Fig. 6**), wodurch die akkumulierten Ausführungszeiten Tpp(m), Tcp(m), Twp(m) und Top(m) erhalten werden. In ähnlicher Weise werden bei der Ende-Verarbeitungsstufe einer Schleife die Ausführungszeit der Schleife und die Ausführungszeit jeder parallel verarbeitenden Bibliotheksroutine, die innerhalb der Schleife aufgerufen wurde, akkumuliert (siehe S15b und S15e in **Fig. 8**), wodurch die akkumulierten Ausführungszeiten Tpl(m), Tcl(m), Twl(m) und Tol(m) für die Schleife erhalten werden. Zusätzlich wird bei der Ende-Verarbeitungsstufe von jeder Schleife die Zahl der Iterationen ebenso akkumuliert. In bezug auf die parallel verarbeitenden Bibliotheksroutinen wird die Ausführungszeit von jeder Routine bei der Ende-Verarbeitungsstufe (siehe S17b in **Fig. 10**) akkumuliert.

Die bei den oben beschriebenen drei Prozeßstufen gesammelten Informationen werden am Ende des Programms summiert und werden dann als eine Profil-Informationsdatei **16** ausgegeben. Zum Wiederauffinden der summierten Informationen aus der Profil-Informationsdatei **16** bietet der Profilanzeigeprozessor **17** Ausführungsprofilsummen dem Anwender dieses Systems an. Um den Anwender mit einer verständlichen und informativen Anzeige zu versehen, verwendet der Profilanzeigeprozessor **17** verschiedene Darstellungsverfahren, die im folgenden beschrieben werden.

Fig. 13 zeigt ein Diagramm, welches ein erstes Beispiel der aus Ausführungsprofilanzeige nach der vorliegenden Erfindung veranschaulicht. In diesem Fall bildet die Prozedur #1 den Hauptkörper des unter Test stehenden parallel verarbeitenden Programms und es werden die Balkengraphiken für die anderen Prozeduren und Schleifen hinsichtlich der Ausführungszeit der Prozedur #1 skaliert (das heißt die Ausführungszeit des gesamten Programms).

Die Ausführungsprofilanzeige, die in **Fig. 13** veranschaulicht ist, enthält Informationen hinsichtlich der Prozeduren #1, #2 usw. und hinsichtlich der Schleifen #1, #2 usw. Jede Balkengraphik ist in vier Teile eingeteilt: die Anwenderprogramm-Ausführungszeit Tu, die Kommunikationsaufgabe-Ausführungszeit Tc, die Synchronisationsaufgabe-Ausführungszeit Tw und eine andere Aufgaben-Ausführungszeit To. Die Aufteilung wird durch den Profilanzeigeprozessor **17** vorgenommen, und zwar im Einklang mit Mittelwerten der angesammelten Ausführungszeiten.

Ferner werden die Ergebnisse der statistischen Analyse für jede Ausführungszeitkategorie auf den Balkengraphiken überlagert, wo die minimalen Werte mit einem Sternchenzeichen "*" wiedergegeben sind, die maximalen Werte mit einem Symbol "x" wiedergegeben sind und die Standardabweichungen mit einem horizontalen Liniensegment wiedergegeben sind, welches zwischen diesen gezeichnet ist.

Eine solche gerade Aus-Anzeige der Ausführungsstatistiken erlaubt es dem Anwender, in einfacher Weise die Beziehung zwischen dem Anwenderprogramm und parallel verarbeitenden Zusätzen, inklusive Kommunika-

tions- und Synchronisationsaufgaben zu unterscheiden. Darüber hinaus kann der Anwender durch das Vorsehen der Maximum-, Minimum- und Standardabweichungssymbole auf einen Blick erkennen, in welcher Weise die Prozentsätze aktuell schwanken. Dieses Merkmal hilft dem Anwender sehr, das Verhalten der Prozeduren und Schleifen zu verstehen, die in einem parallel verarbeitenden Programm enthalten sind, so daß dieser die Möglichkeit erhält, zu beobachten, ob das Programm richtig arbeitet oder nicht und die Entscheidung treffen kann, ob weitere Programm-Modifikationen erforderlich sind oder nicht, um eine Qualitätsverbesserung zu erreichen.

Fig. 14 zeigt ein Diagramm, welches ein zweites Beispiel der Ausführungsprofilanzeige wiedergibt. Im Gegensatz zu dem ersten in Fig. 13 gezeigten Beispiel, werden bei dem Verfahren nach dem zweiten Beispiel alle Prozeduren und Schleifen in den individuellen Zeitmaßstäben sichtbar gemacht, ungeachtet von deren absoluten Ausführungszeiten. Das heißt, die Längen der Balkengraphiken sind nicht proportional zu deren jeweiligen Ausführungszeitlängen. Die Graphiken in Fig. 14 erlauben es jedoch, daß der Anwender auf einfache Weise die Prozentsätze der Programmkomponenten verstehen kann (das heißt Anwenderprogramm Tu, Kommunikationsaufgabe Tc, Synchronisationsaufgabe Tw und andere Aufgaben To), die in jeder Prozedur oder Schleife involviert sind.

Fig. 15 zeigt ein drittes Beispiel einer Ausführungsprofilanzeige, bei der zwei Prozeduren #1 und #2 in Form einer Kreisgraphik wiedergegeben sind. Kreisgraphiken erlauben im allgemeinen, daß der Anwender die Beziehung von Komponenten zu einer Gesamtheit einfacher vergleichen kann. Bei diesem dritten Anzeigeverfahren zeigt ein gesamter Kreis von 360 Grad die gesamte akkumulierte Ausführungszeit einer Prozedur oder Schleife und vier Sektoren zeigen die Prozentsätze der Anwenderprogramm-Ausführungszeit (Tu), der Kommunikationsaufgabe-Ausführungszeit (Tc), der Synchronisationsaufgabe-Ausführungszeit (Tw) und der anderen Aufgabe-Ausführungszeit (To) an.

Fig. 16 zeigt ein viertes Beispiel einer Ausführungsprofilanzeige, bei der das Ausführungsprofil in Form einer Radarkarte wiedergegeben ist. Die Radarkarte in Fig. 16 besitzt vier Achsen entsprechend den vier Programmkomponenten. Die akkumulierten Ausführungszeitwerte (Tu, Tc, Tw und To) der Programmkomponenten sind auf deren jeweiligen Achsen aufgetragen und die aufgetragenen Punkte sind miteinander verbunden, um ein Polygon zu bilden. Aus der Gestalt dieses Polygons kann der Anwender intuitiv die Eigenschaften der Prozedur oder Schleife verstehen, die in einer Radarkarte profiliert sind.

Ähnlich den anderen Anzeigeverfahren, die soweit beschrieben wurden, umfaßt die Radarkarte, die in Fig. 16 veranschaulicht ist, Informationen hinsichtlich der Minimal- und Maximalwerte und der Standardabweichung der akkumulierten Ausführungszeit. Dieses Merkmal ermöglicht es dem Anwender, das Verhalten jeder Programmkomponente in einem detaillierteren Wert zu verstehen.

Fig. 17 zeigt ein fünftes Beispiel einer Ausführungsprofilanzeige gemäß der vorliegenden Erfindung. Ein Ausführungsprofilfenster 50 besteht aus vier Abschnitten 50a bis 50d. Der erste Abschnitt 50a enthält die Legende der Balkengraphiken, die in den anderen Abschnitten des Fensters 50 dargestellt sind. Spezieller gesagt, definiert die Legende die Farbklassifizierung für vier verschiedene Programmkomponenten, die bezeichnet sind mit:

Anwender: Anwenderprogramm.
Com: Kommunikationsaufgabe-Routine.
Sync: Synchronisationsaufgabe-Routine.
Runlib und Gop: andere Routinen.

Das Ausführungsprofilfenster 50 bei dem fünften Beispiel verwendet diese Farbklassifizierung zusammen mit dem Darstellungsverfahren wie in Fig. 13.

Der zweite Abschnitt 50b des Ausführungsprofilfensters 50 zeigt ein prozedur-gestütztes Ausführungsprofil, welches die akkumulierten Ausführungszeiten von mehreren Prozeduren, wie beispielsweise "MAIN", "calc1", "calc2", "cputim" und "inital", aufsummiert.

Der dritte Abschnitt 50c des Ausführungsprofilfensters 50 zeigt ein fang-gestütztes (caller-based) Ausführungsprofil, welches die akkumulierte Ausführungszeit von jeder Prozedur zusammen mit dem Namen von dessen Fang-Routine (caller routine) zusammenfaßt. Beispielsweise liest sich die oberste Zeile dieses Abschnitts wie folgt:

calc1 <10,292s> MAIN <UNSPECIFIED

worin das links bezeichnete "calc1" der Name der Prozedur ist, die durch eine Fang-Routine "MAIN" gerufen wurde. Das am weitesten rechts gelegene Zeichen" <UNSPECIFIED" bedeutet, daß die Fang-Routine nicht ihren eigenen Programmnamen deklariert und der inhärente Name "MAIN" durch das Analysierersystem verwendet wird. Die Zeile zeigt auch an, daß die akkumulierte Ausführungszeit dieser Prozedur "calc1" gleich 10,292 Sekunden beträgt.

Die Balkengraphik für diese "calc1"-Prozedur ist farb-klassifiziert gemäß der Legende, die in dem ersten Abschnitt 50a definiert ist, wobei vier Teile der Balkengraphik jeweils die akkumulierten Ausführungszeiten Tu, Tc, Tw und To zeigt, und zwar zusammen mit der Anzeige des Minimums, Maximums und der Standardabweichung.

Der vierte Abschnitt 50d des Ausführungsprofilfensters 50 zeigt ein rufer-gestütztes Ausführungsprofil. Beispielsweise liest sich die oberste Zeile des Abschnitts 50d wie folgt:

MAIN <UNSPECIFIED inital

worin die Angabe "MAIN <UNSPECIFIED" den Namen einer Rufer-Routine zeigt und die andere Aufschrift

"inital" den Namen einer gerufenen Prozedur anzeigt. Die Balkengraphik für diese "inital"-Prozedur ist farb-klassifiziert gemäß der in dem ersten Abschnitt 50a definierten Legende. Sie besteht aus vier Teilen, die jeweils die akkumulierten Ausführungszeiten Tu, Tc, Tw und To als auch deren Minimum- und Maximumwerte und Standardabweichung mit den zuvor erwähnten Symbolen zeigen.

Fig. 18 zeigte ein sechstes Beispiel der Ausführungsprofildarstellung gemäß der vorliegenden Erfindung. Ein Ausführungsprofilfenster 60 besteht aus vier Abschnitten 60a bis 60d. Der erste Abschnitt 60a liefert die Legende von Balkengraphiken, die in den anderen Abschnitten des Fensters 60 dargestellt werden und spezieller definiert ist die Farbklassifikation für vier Programmkomponenten, die bezeichnet sind mit:

Anwender: Anwenderprogramm.

Com: Kommunikationsaufgabe-Routine.

Sync: Synchronisationsaufgabe-Routine.

Runlib und Gop: andere Routinen.

Die Legende umfaßt auch die Farbklassifizierung für Verzweigungsbedingungen, die enthält "Wahr"- und "Falsch"-Bedingungen für logische Operationen und "negative", "Null-" und "positive" Bedingungen für arithmetische Operationen.

Der zweite Abschnitt 60b des Ausführungsprofilfensters 60 zeigt ein Ausführungsprofil, welches die akkumulierten Ausführungszeiten für mehrere Schleifen zusammenfaßt.

Der dritte Abschnitt 60c des Ausführungsprofilfensters 50 zeigt ein Ausführungsprofil, welches die Verzweigungsbedingungen zusammenfaßt. Die obere Balkengraphik in dem dritten Abschnitt 60c veranschaulicht die Prozentsätze von "Wahr"-Bedingungen und "Falsch"-Bedingungen, die tatsächlich als ein Ergebnis der logischen Operationen aufgetreten sind. Die hintere Balkengraphik veranschaulicht die Prozentsätze von "positiven", "Null-" und "negativen" Bedingungen, die tatsächlich als ein Ergebnis der arithmetischen Operationen aufgetreten sind.

Der vierte Abschnitt 16d des Ausführungsprofilfensters 60 liefert ein gerufenen-gestütztes (caller-based) Ausführungsprofil, welches das gleiche ist wie bei dem dritten Abschnitt 50c in Fig. 17.

Zusammenfassend für die Fig. 12 bis 17 gibt die vorliegende Ausführungsform die Ergebnisse der Leistungsanalyse wieder, und zwar unter Verwendung einer Vielfalt von Darstellungsverfahren. Neben der Möglichkeit, die mittlere Ausführungszeit von jeder Subjekt-Routine darstellen zu können, zeigt das vorliegende Analysierersystem die Maximum-, Minimum- und Standardabweichung der Ausführungszeiten, wodurch der Anwender die Möglichkeit erhält, mit einem Blick zu erkennen, wie die Prozentsätze tatsächlich schwanken.

Bei der vorliegenden Ausführungsform analysiert das Leistungsanalysierersystem das Parallelrechnersystem 11 (siehe Fig. 1), welches aus vielfachen Prozessoren #1 bis #n besteht. Jedoch kann die vorliegende Erfindung auch auf ein einzelnes Prozessorsystem angewandt werden, welches vielfache Aufgaben ausführt, und zwar durch Verwendung eines Zeitteilungsverfahrens. In diesem Fall sammeln mehrere geeignete Laufzeit-Subroutinen Ausführungsdaten für einzelne Prozesse oder Module, welche virtuell parallel laufen und das Leistungsanalysierersystem summiert die gesammelten Daten auf und stellt die Ergebnisse als Ausführungsprofile dar. Wie bei der vorliegenden Ausführungsform zeigt das Ausführungsprofil, welches auf einem Bildschirm dargestellt wird, mittlere, Minimum- und Maximum-Ausführungszeiten und die Standardabweichung von dem Mittelwert hinsichtlich der Subjektprozesse oder Module.

Die vorangegangene Beschreibung dient lediglich der Veranschaulichung der Prinzipien der vorliegenden Erfindung. Da ferner vielfältige Modifikationen und Änderungen für den Fachmann offensichtlich sind, ist nicht beabsichtigt, die Erfindung auf die exakte Konstruktion und Anwendungen zu beschränken, die beschrieben wurden, und demzufolge fallen alle geeigneten Modifikationen und äquivalenten Ausführungen in den Rahmen der Erfindung gemäß den anhängenden Ansprüchen und deren Äquivalente.

Patentansprüche

1. Verfahren zur sichtbaren Darstellung von Ergebnissen einer Leistungsüberwachung und -analyse für ein Parallelrechnersystem, bei dem eine Vielzahl von Prozessoren ein parallel verarbeitendes Programm ausführen, welches aus einer Vielzahl von Routinen zusammengesetzt ist, wobei das Verfahren die folgenden Schritte umfaßt:

(a) Sammeln von Informationen hinsichtlich der Ausführungszeit von jeder Routine, die gleichlaufend durch eine Vielzahl von Prozessoren ausgeführt wird, unter Berücksichtigung der Klassifizierung der Routinen;

(b) Ableiten eines Maximalwertes, eines Mittelwertes, eines Minimalwertes und einer Standardabweichung der Ausführungszeit von jeder Routine basierend auf den Informationen, die für die Vielzahl der Prozessoren gesammelt wurden; und

(c) graphisches Darstellen der Ausführungsprofil-Information, welche den Maximalwert, den Mittelwert und den Minimalwert und die Standardabweichung der Ausführungszeit jeder Routine enthält.

2. Verfahren nach Anspruch 1, bei dem bei dem Schritt (a) die Ausführungszeit von jeder Routine akkumuliert wird, jedesmal, wenn die Routine aufgerufen und ausgeführt wird.

3. Verfahren nach Anspruch 1, bei dem bei dem Schritt (a) die Ausführungszeit von jeder Routine gesammelt wird, die als ein Subjekt der Profilierung bezeichnet wurde.

4. Verfahren nach Anspruch 1, bei dem bei dem Schritt (a) die Routinen in ein Anwenderprogramm, Kommunikationsaufgaben, Synchronisationsaufgaben und andere Aufgaben klassifiziert werden.

5. Verfahren nach Anspruch 3, bei dem das Subjekt der Profilierung Prozeduren, Schleifen und parallel verarbeitende Bibliotheksroutinen enthält.

6. Verfahren nach Anspruch 1, bei dem bei dem Schritt (b) ferner die Standardabweichung der Ausführungs-

zeit von jeder Routine zusätzlich zu dem Maximalwert, dem Mittelwert und dem Minimalwert berechnet wird.

7. Verfahren nach Anspruch 1, bei dem bei dem Schritt (c) Rufer-Routinen und/oder Gerufener-Routinen aus den Routinen in dem parallel verarbeitenden Programm extrahiert werden und die Ausführungsprofilinformation, welche zu den extrahierten Routinen gehört, dargestellt wird.

8. Verfahren nach Anspruch 1, bei dem bei dem Schritt (c) eine Balkengraphik verwendet wird, um graphisch die Ausführungsprofilinformation von jeder Routine darzustellen.

9. Verfahren nach Anspruch 1, bei dem bei dem Schritt (c) eine Kreisgraphik verwendet wird, um graphisch die Ausführungsprofilinformation von jeder Routine darzustellen.

10. Verfahren nach Anspruch 1, bei dem bei dem Schritt (c) eine Radarkarte verwendet wird, um die Ausführungsprofilinformation von jeder Routine graphisch darzustellen.

11. Verfahren nach Anspruch 1, welches ferner die folgenden Schritte umfaßt:

(d) Annehmen einer Bezeichnung eines Punktes der Ausführungsprofilinformation, die bei dem Schritt (c) dargestellt wird, und

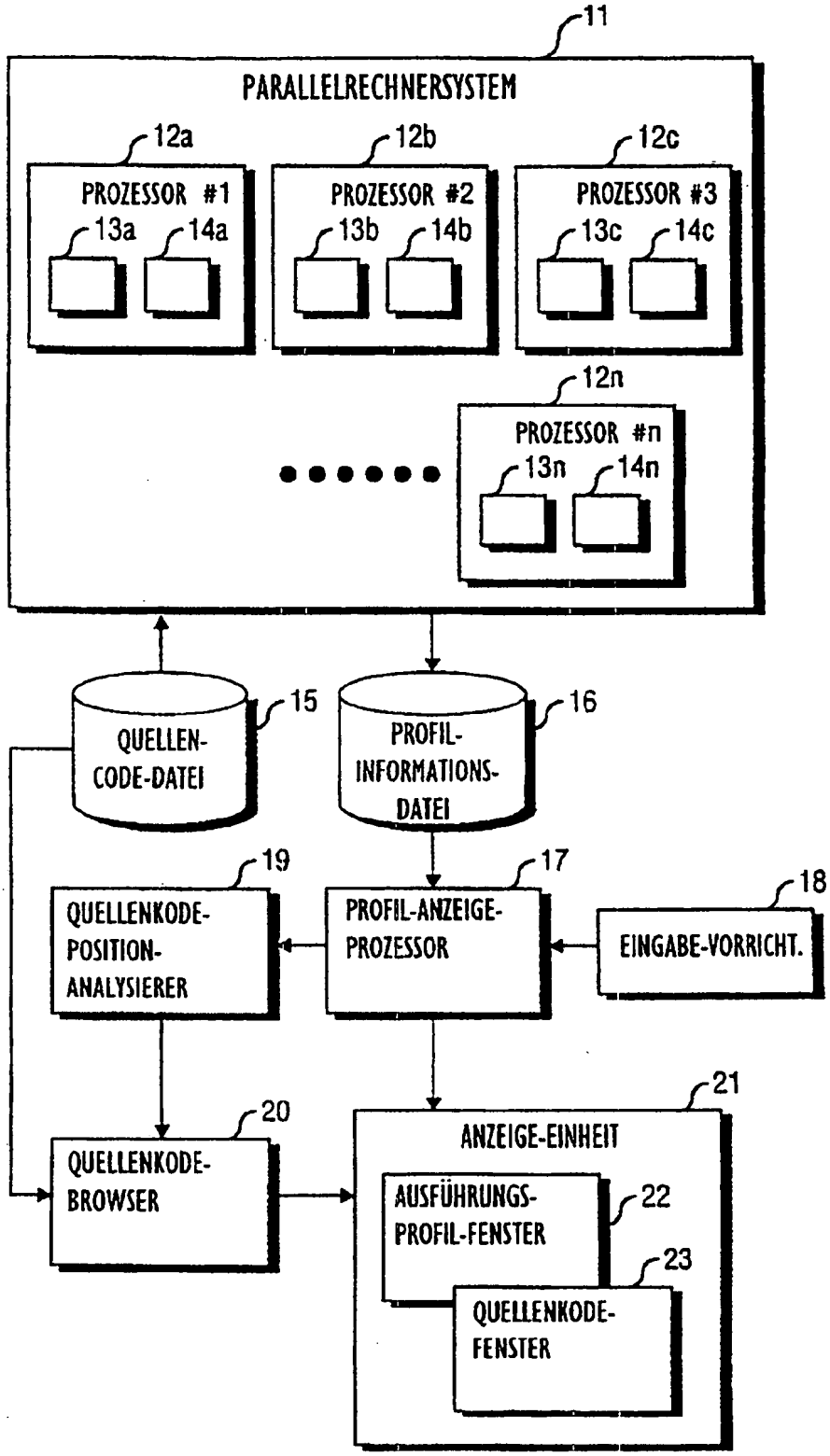
(e) Darstellen eines Teiles des Quellencodes des parallel verarbeitenden Programms, welches dem bezeichneten Punkt der Ausführungsprofilinformation entspricht.

12. Ein computer-lesbares Medium, welches mit einem Computer-Programm codiert ist, bei dem das Verfahren nach Anspruch 1 implementiert ist.

Hierzu 18 Seite(n) Zeichnungen

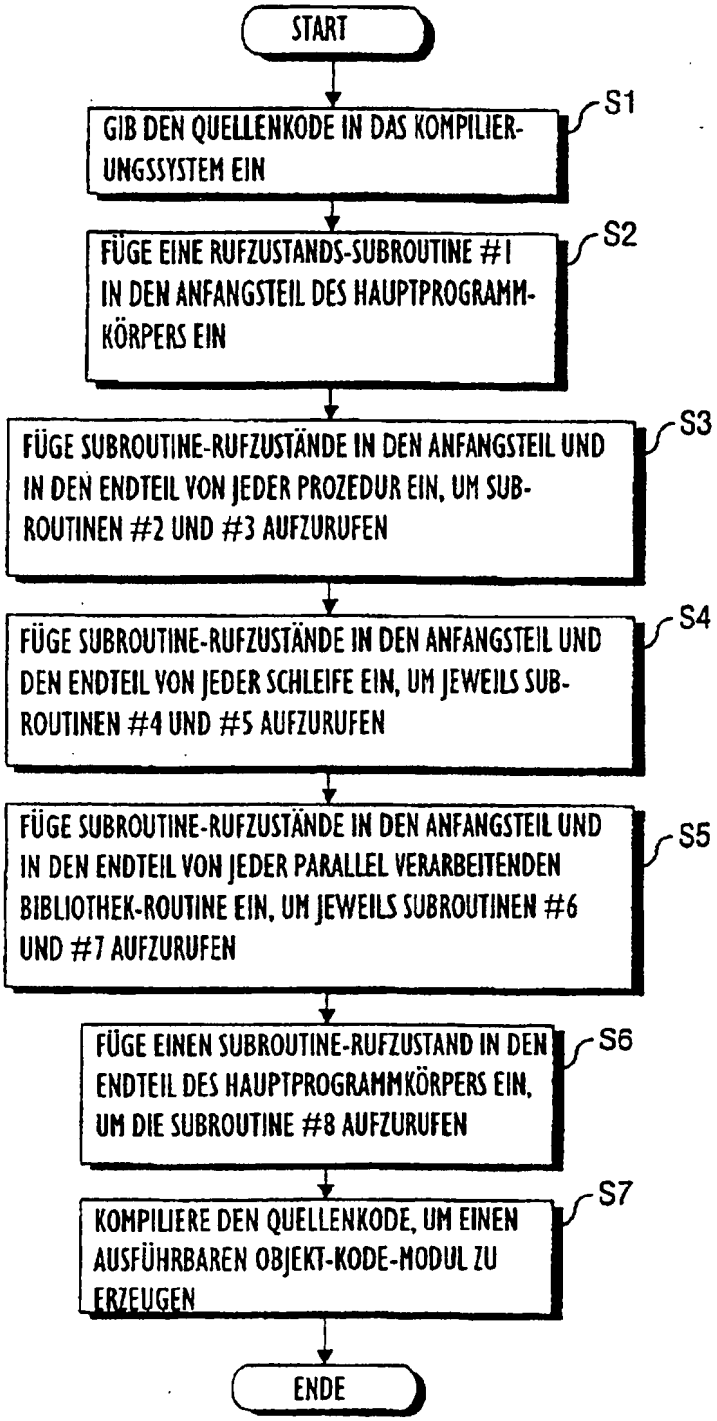
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 1



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 2



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

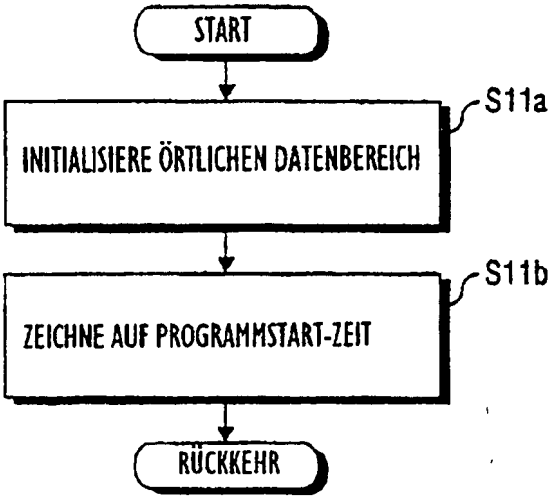
FIG. 3

30

LAUFZEIT-SUBROUTINE	AUFGERUFEN BEI	PRIMÄRE FUNKTIONEN
#1	BEGINN DES PROGRAMMS	INITIALISIERE ORTLICHEN DATEN-BEREICH
#2	BEGINN DER PROZEDUR	AUFZEICHNE AUSFÜHRUNGS-STARTZEIT VON JEDER PROZEDUR
#3	ENDE DER PROZEDUR	AKKUMULIERE DIE AUSFÜHRUNGSZEIT VON JEDER PROZEDUR
#4	ANFANG DER SCHLEIFE	AUFZEICHNE DIE AUSFÜHRUNGS-STARTZEIT VON JEDER SCHLEIFE
#5	ENDE DER SCHLEIFE	AKKUMULIERE AUSFÜHRUNGSZEIT VON JEDER SCHLEIFE
#6	ANFANG DER PARALLEL VER-ARBEITENDEN BIBLIOTHEK-ROUTINE	AUFZEICHNE AUSFÜHRUNGS-STARTZEIT VON JEDER ROUTINE
#7	ENDE DER PARALLEL VER-ARBEITENDEN BIBLIOTHEK-ROUTINE	AKKUMULIERE AUSFÜHRUNGSZEIT VON JEDER ROUTINE
#8	ENDE DES PROGRAMMS	AUFSUMMIERE DIE AUSFUHRUNGSZEITDATEN UND GBE AUS PROFIL-INFO-DATEI

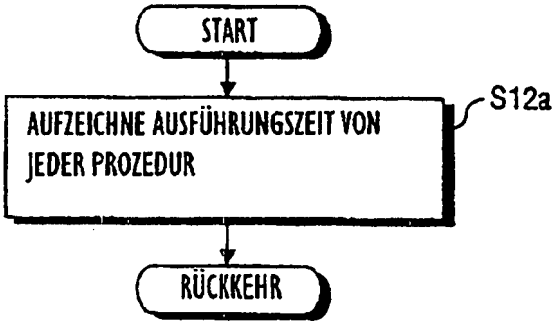
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 4



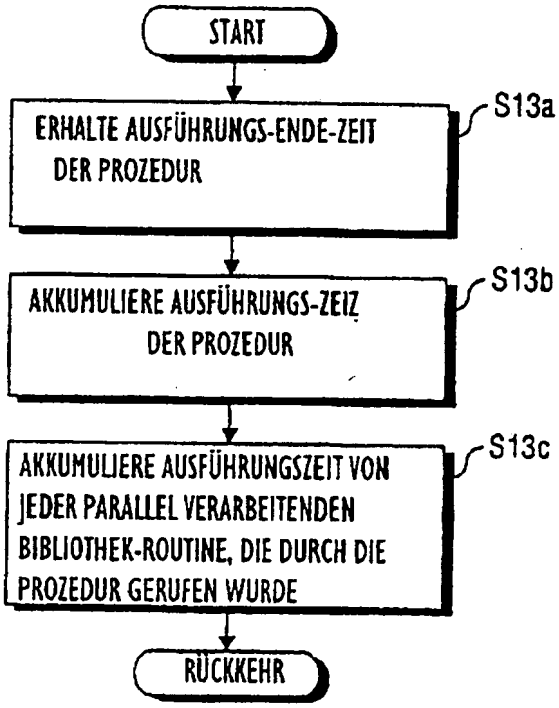
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 5



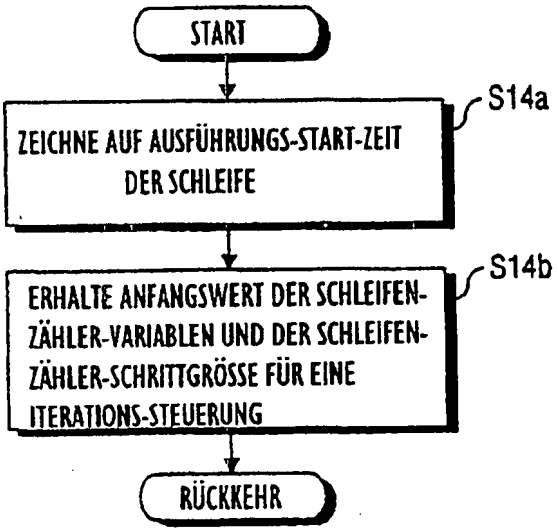
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 6



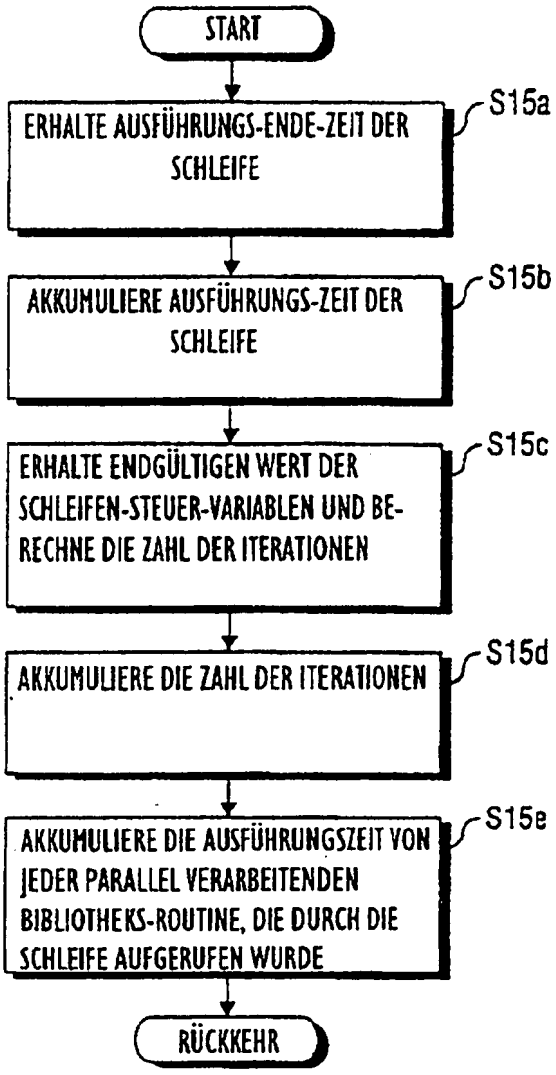
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 7



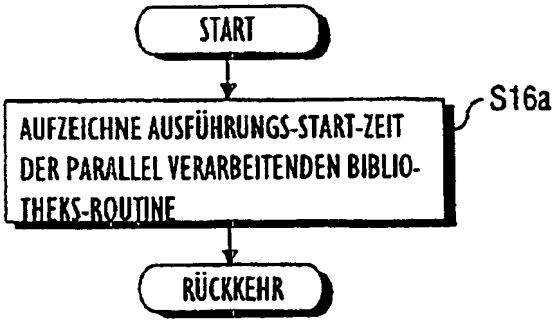
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 8



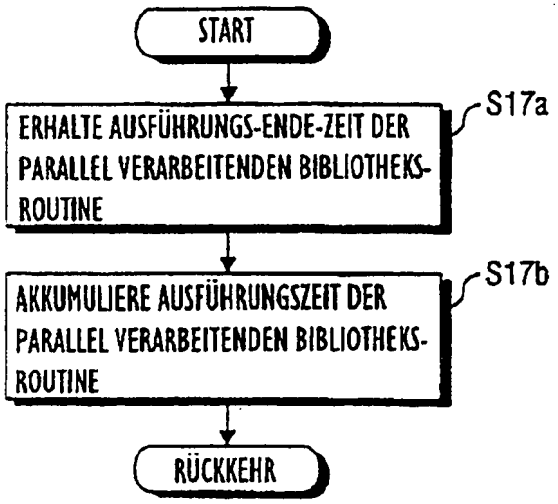
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 9



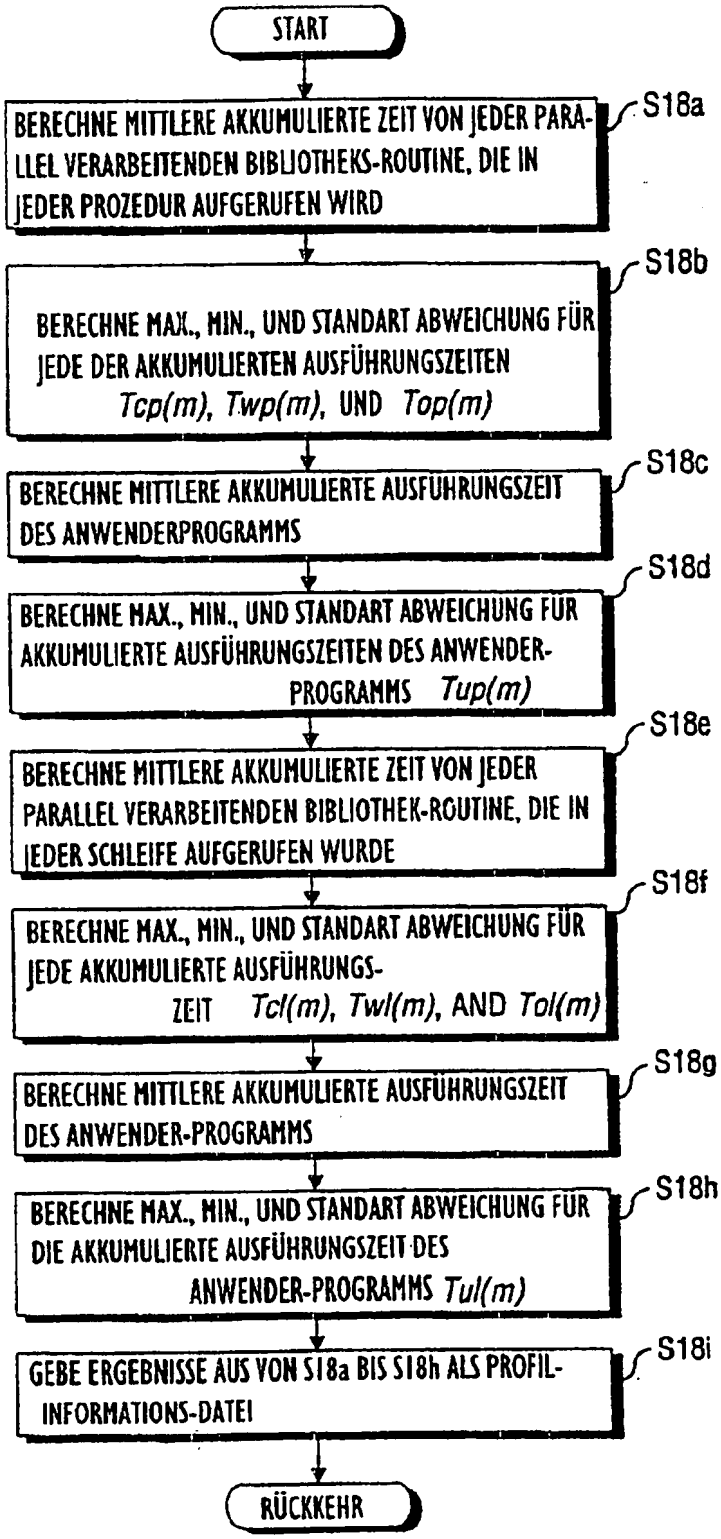
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 10



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 11



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

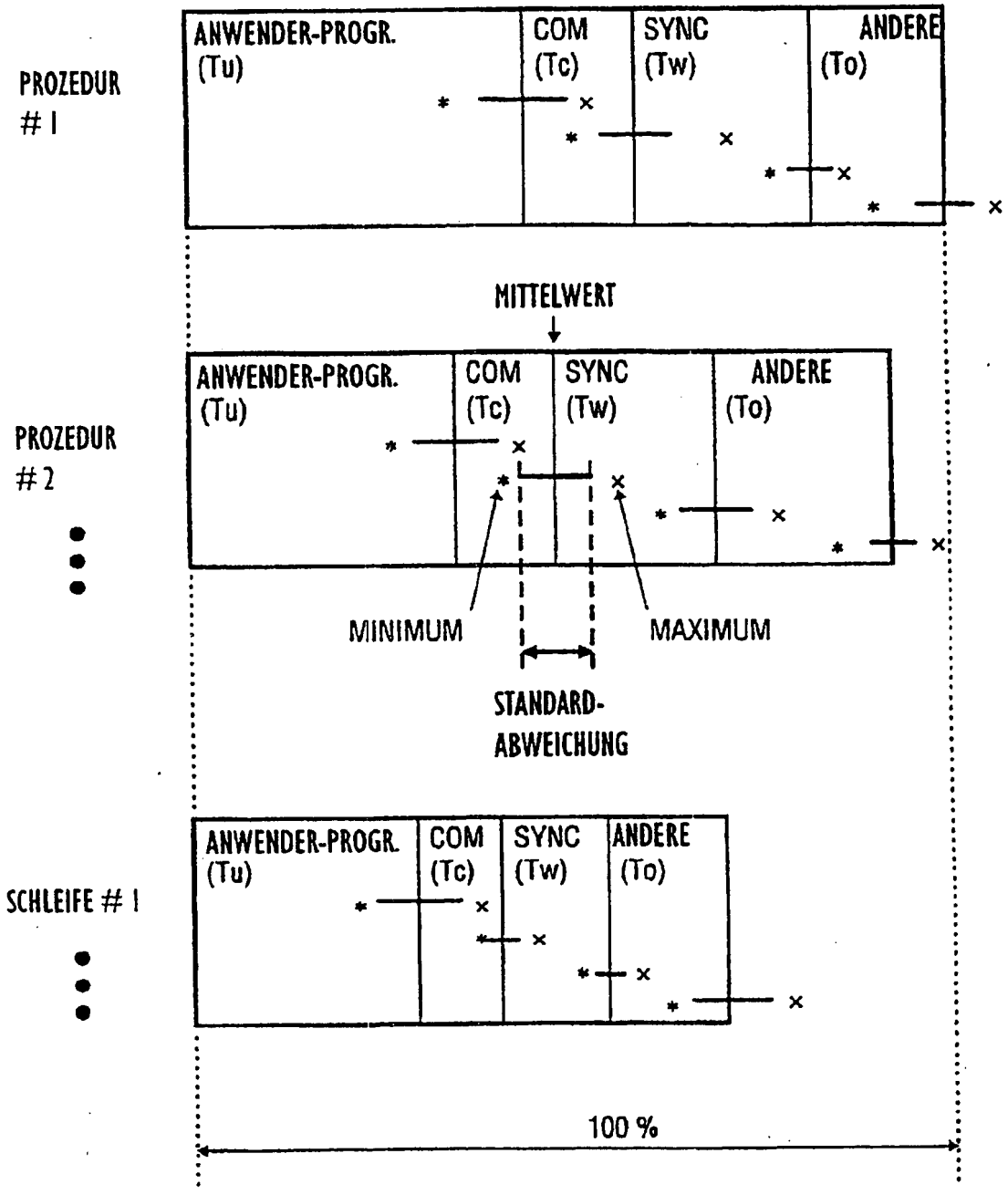
FIG. 12

40

SUBJEKT DER PROFILIERUNG	VORBEREITUNGS-PROZESS-STUFE	ANLAUF-PROZESS-STUFE	BEENDIGUNGS-PROZESS-STUFE
PROZEDUR		• AUSFÜHRUNGS-START-ZEIT	• AKKUMULIERE AUSFÜHRUNGS-ZEIT VON JEDER PROZEDUR • AKKUMULIERE AUSFÜHRUNGS-ZEIT VON JEDER PARALLEL VERARBEITENDEN BIBLIOTHEKS-ROUTINE, DIE DURCH DIE PROZEDUR AUFGERUFEN WURDE
SCHLEIFE		• AUSFÜHRUNGS-START-ZEIT • SCHLEIFEN-ZÄHLER-ANFANGSWERT • SCHLEIFEN-ZÄHLER-SCHRITT-GRÖSSE	• AKKUMULIERE AUSFÜHRUNGS-ZEIT VON JEDER SCHLEIFE • AKKUMULIERE DIE ZAHL DER ITERATIONEN • AKKUMULIERE AUSFÜHRUNGS-ZEIT VON JEDER PARALLEL VERARBEITENDEN BIBLIOTHEKS-ROUTINE, DIE DURCH DIE PROZEDUR AUFGERUFEN WURDE
PARALLEL VERARBEITENDE BIBLIOTHEKS-ROUTINE		• AUSFÜHRUNGS-START-ZEIT	• AKKUMULIERE AUSFÜHRUNGS-ZEIT VON JEDER ROUTINE

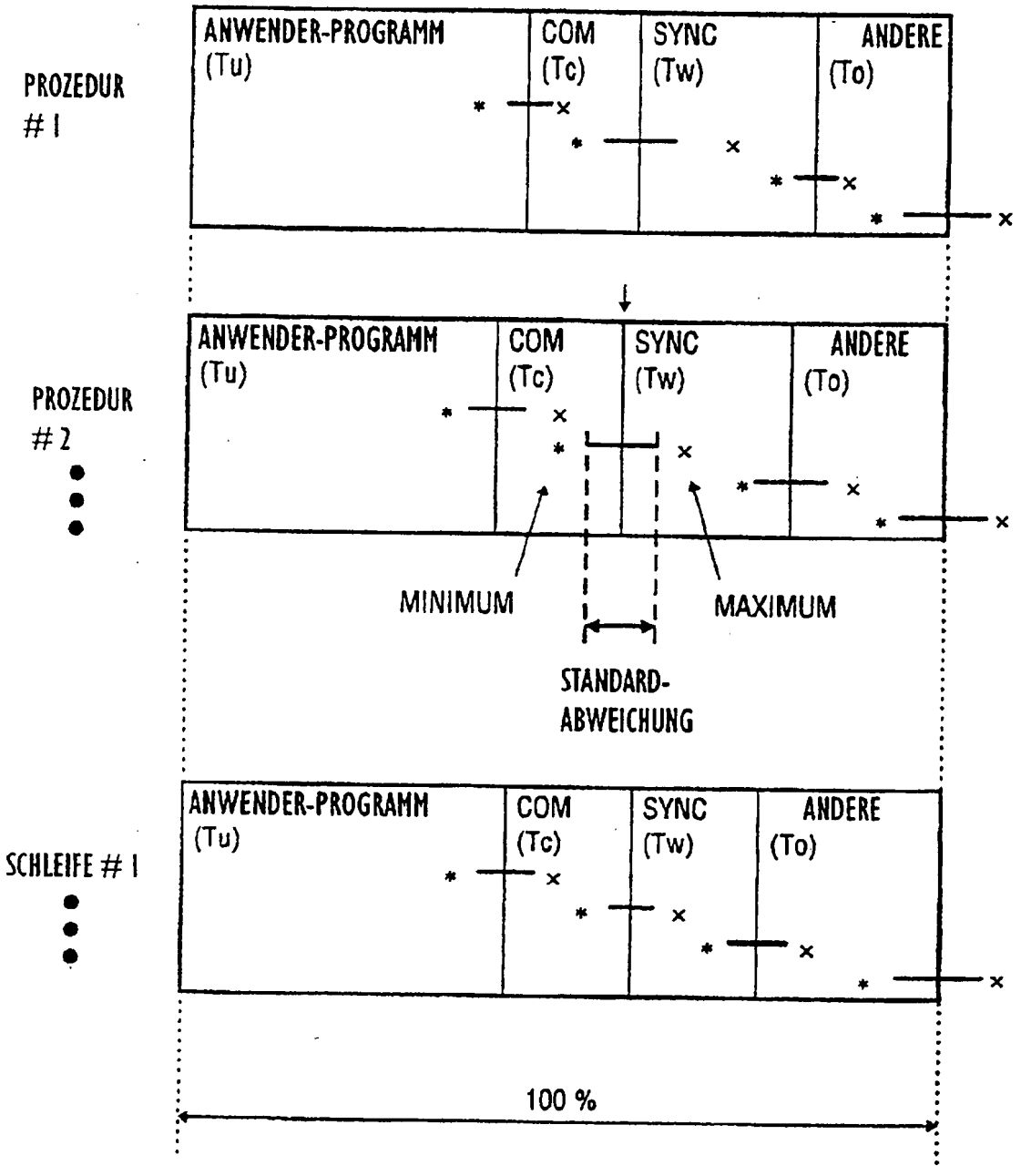
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 13



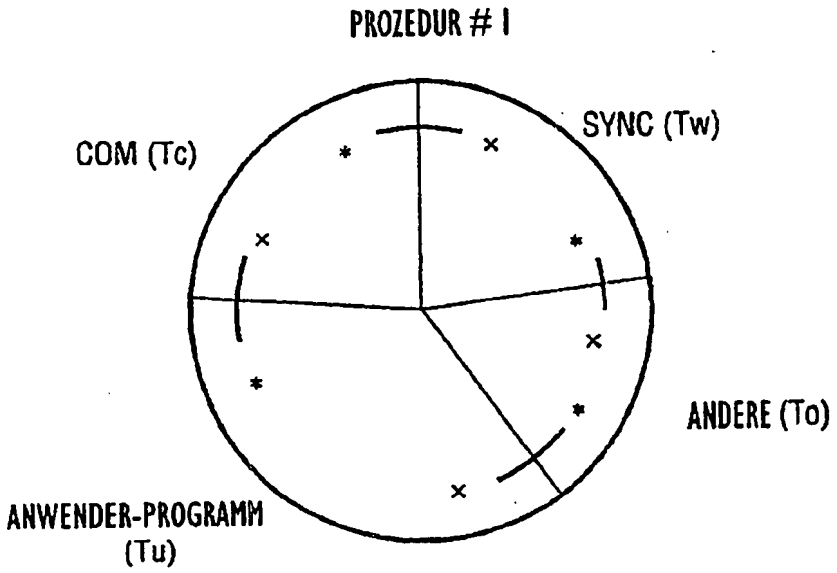
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

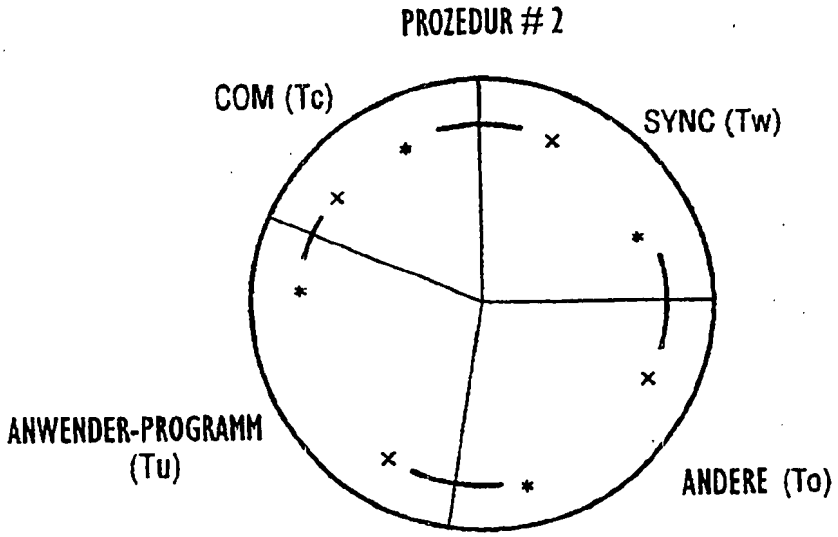
FIG. 14



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

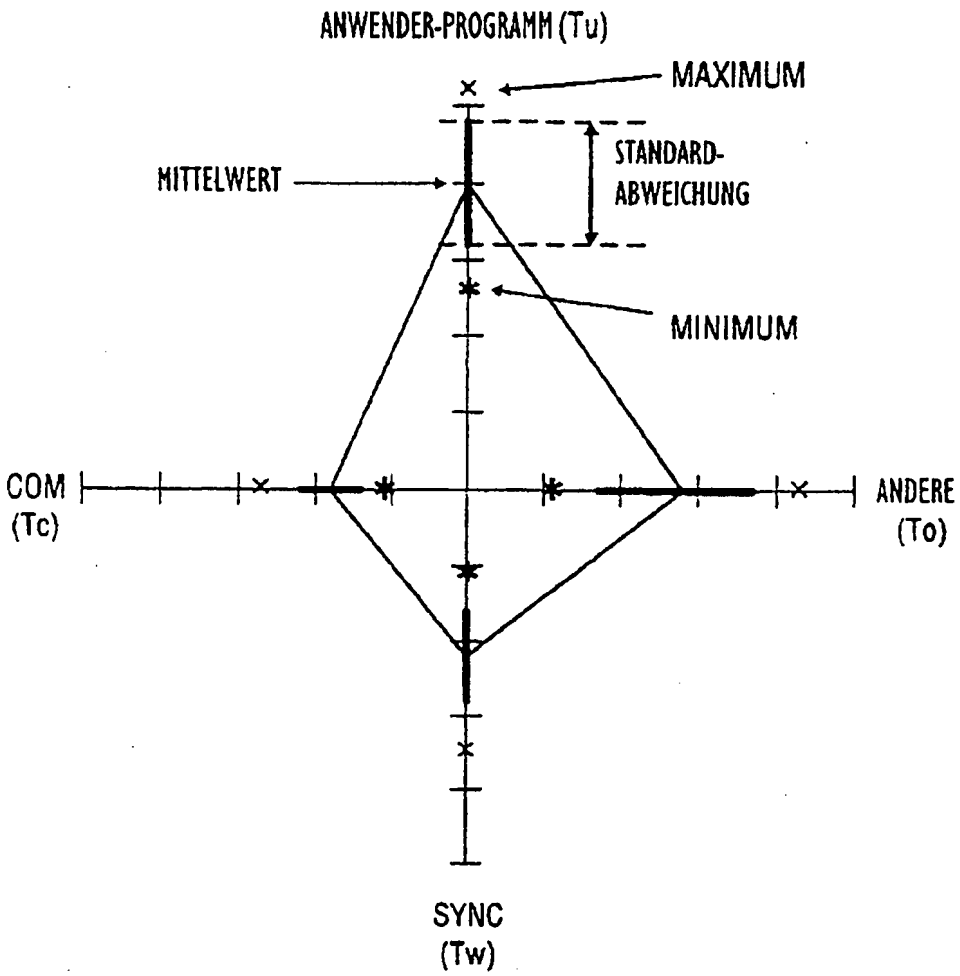
FIG. 15





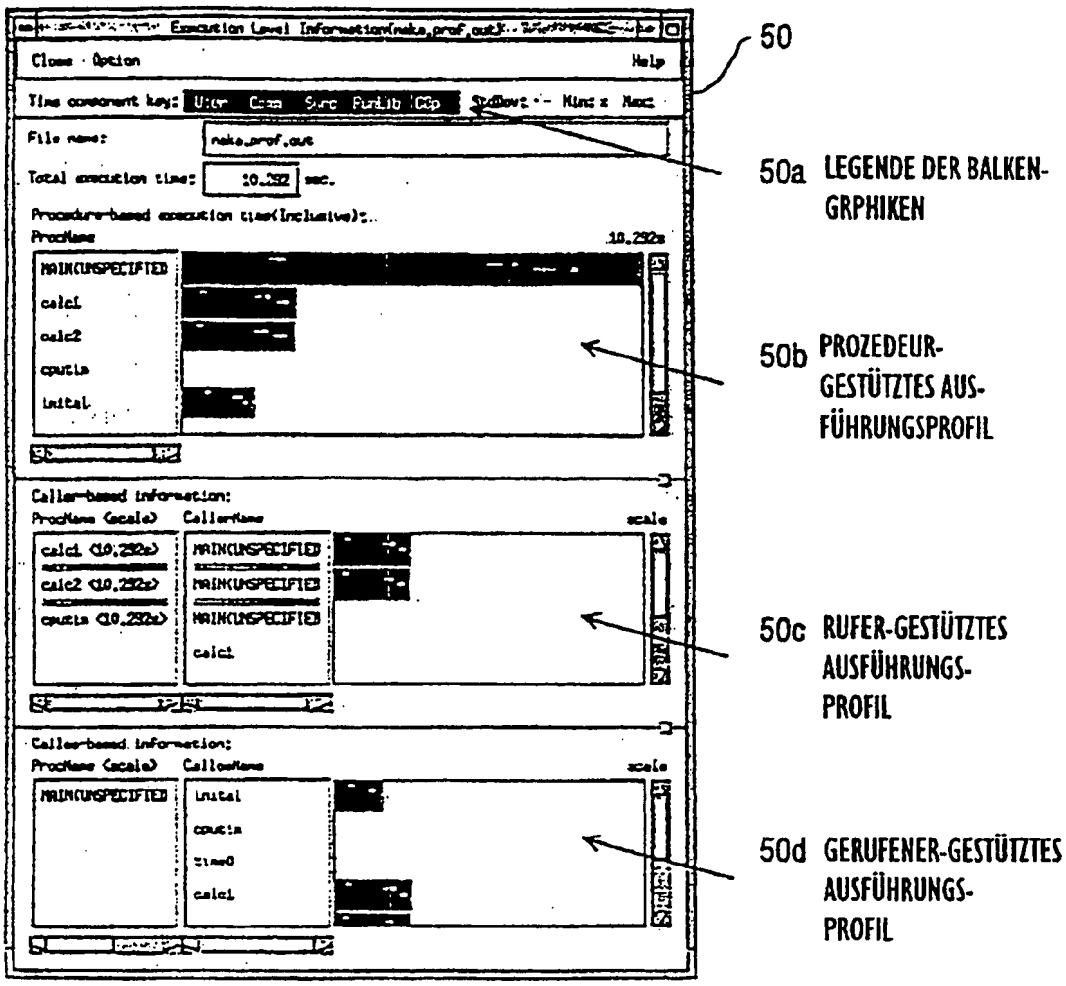
Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 16



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 17



Nummer: DE 197 10 252 A1
Int. Cl.[6]: G 06 F 11/34
Offenlegungstag: 26. Februar 1998

FIG. 18